MTAT.03.227 Machine Learning

Model-based Clustering

Sven Laur
University of Tartu

Ancestral reconstruction

General formulation. Given a set of objects that are created by noisy reproduction procedure find out which of them is a true original.

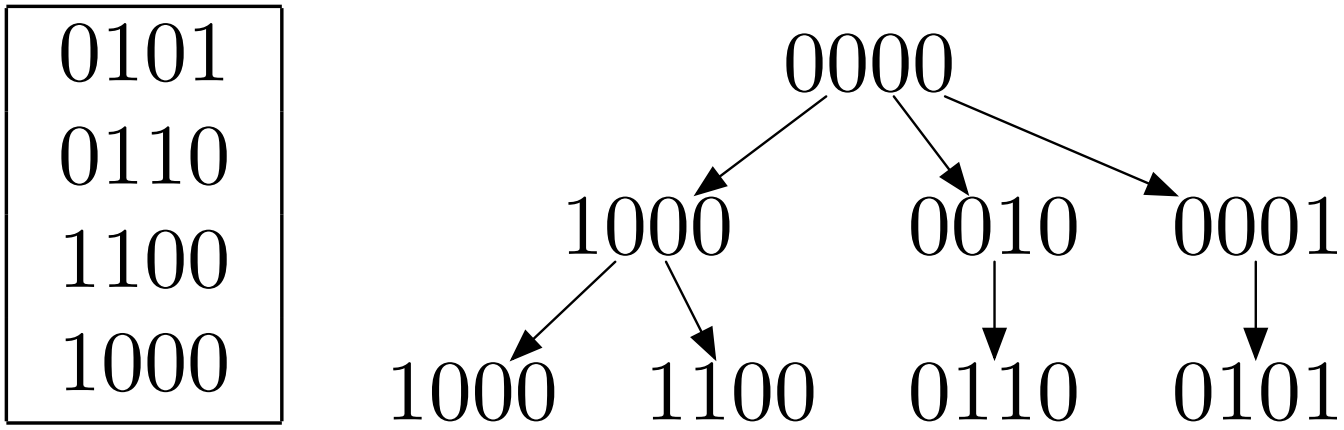
- ◇ Find out how species have been evolving using DNA samples
- ◇ Find out which of the ancient manuscripts is the original
- ◇ Find out the source of a gossip and evaluation of internet memes
- ◇ Find out how academic texts are plagiarised

General principle. Errors made by the coping mechanism are very likely to be propagated to the further copies.

- ▷ The copy without errors (*mutations*) must be the true original.
- ▷ To reconstruct evolutionary tree we must back-track all mutations.

Illustrative example

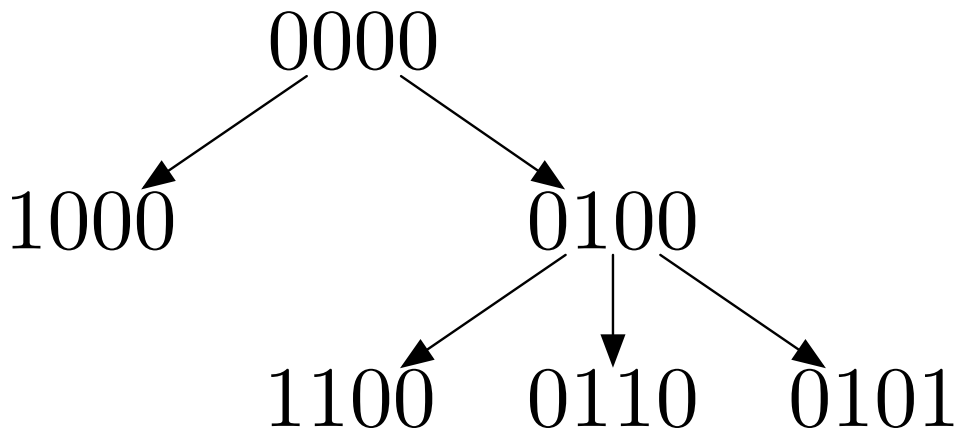
Assume that we know the true original and there are four possible errors in documents. Then we can construct error vectors and define a set of plausible evolutionary explanations. One of them is depicted below.



In most cases the copying procedure is almost perfect and occurrence of a single error is much more likely than the occurrence of two or more errors.

Illustrative example continued

The evolutionary tree presented in the previous slide contains six mutations, whereas more plausible explanation below contains only four mutations.



Naive mutation model

- ▷ All sites are independently flipped with probability p
- ▷ A child is obtained form a parent through a single copying operation
- ▷ A copying operations are independent form each other

Correponding mathematical model. Let $h(\boldsymbol{u}, \boldsymbol{v}) = \#\{i : u_i \neq v_i\}$ and n the length of vectors. Then the probability that $\boldsymbol{v}$ is child of $\boldsymbol{u}$ is

$$\Pr[\boldsymbol{u} \to \boldsymbol{v}] = (1-p)^{n-h(\boldsymbol{u},\boldsymbol{v})} p^{h(\boldsymbol{u},\boldsymbol{v})}$$

The probability of the entire tree is the product of edge probabilities:

$$\Pr[\mathcal{T}] = \prod_{\boldsymbol{u}\to\boldsymbol{v}} \Pr[\boldsymbol{u} \to \boldsymbol{v}] = \prod_{\boldsymbol{u}\to\boldsymbol{v}} (1-p)^n \left(\frac{p}{1-p}\right)^{h(\boldsymbol{u},\boldsymbol{v})}$$

Corresponding minimisation task

Let E denote the set of edges in the evolutionary tree. Then we can express

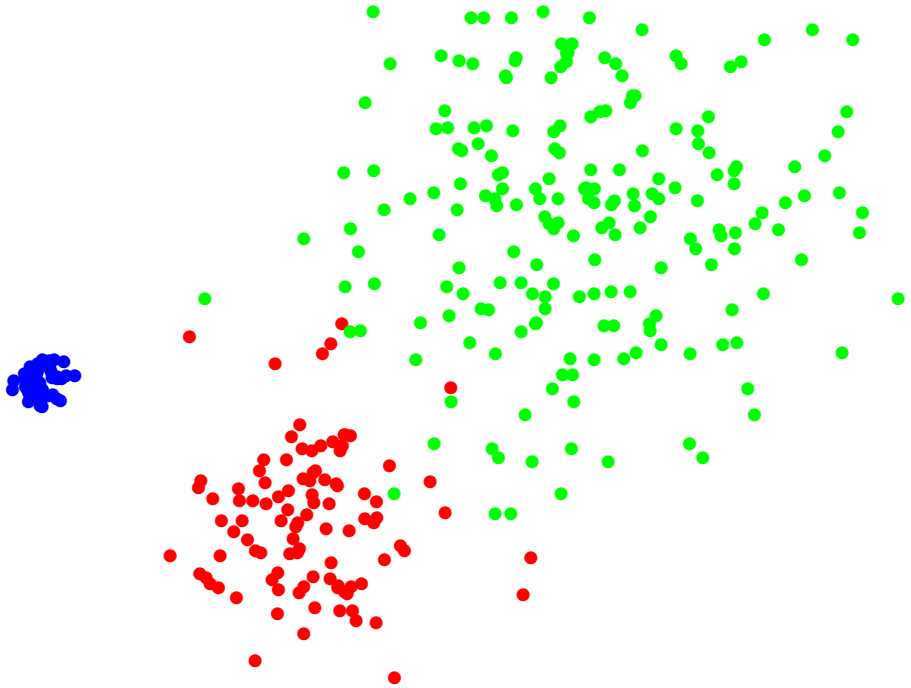
$$\log \Pr[\mathcal{T}] = |E| \cdot n \cdot \log(1-p) + \log\left(\frac{p}{1-p}\right) \cdot \sum_{\boldsymbol{u}\to\boldsymbol{v}} h(\boldsymbol{u},\boldsymbol{v})$$

By dividing log-likelihood with a constant $n \cdot \log(1-p)$ we get a simpler minimisation goal:

$$|E| + \tau(p) \cdot \sum_{\boldsymbol{u}\to\boldsymbol{v}} h(\boldsymbol{u},\boldsymbol{v}) \to \min$$

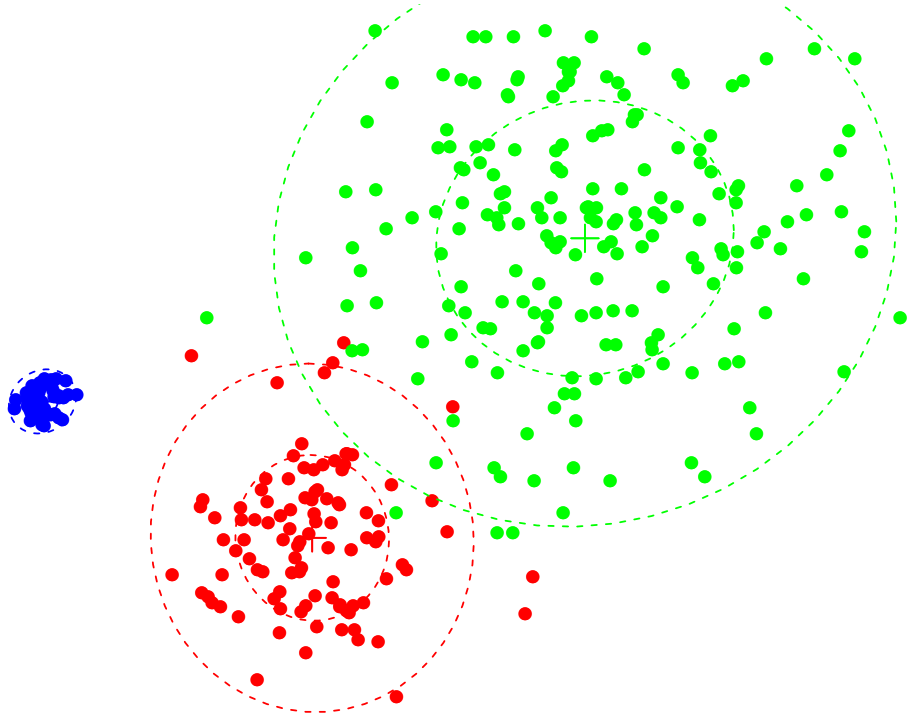
which implies that for trees with equal size we should take the one with fewer changes. For different tree sizes, the choice depends on $\tau(p)$ value.

Target reconstruction in shooting



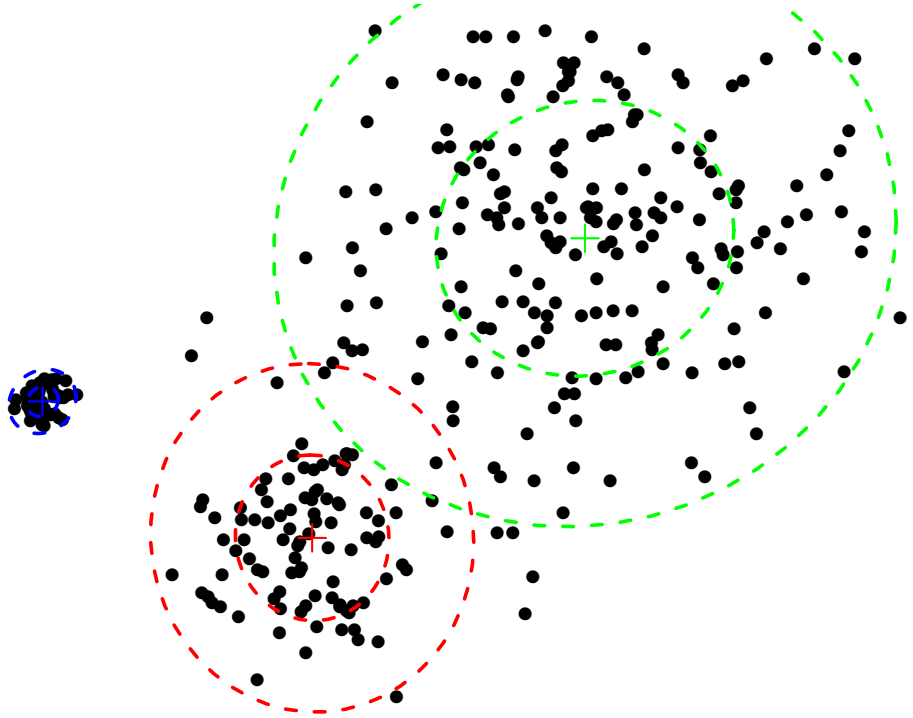
Find the true target for each shooter given that errors in all directions are equiprobable and distributed according to normal distribution.

Target reconstruction in shooting



Reconstruction of the true targets is straightforward when we know the origin of individual shots. We discuss later how to do it exactly.

Target reconstruction in shooting



Reconstruction of the true targets is much harder if we have to guess the origin of shots by ourselves. This task is known as clustering.

Naive probabilistic model

- There are k different data sources $\mathcal{D}_1, \ldots, \mathcal{D}_k$
- Each data source has a fixed centre $\boldsymbol{\mu}_j \in \mathbb{R}^d$
- For each data point noise $\varepsilon_t \leftarrow \mathcal{N}(0, \sigma)$ for each coordinate
- To sample data from the distribution $\mathcal{D}_j$, we output $\boldsymbol{x} = \boldsymbol{\mu}_j + \boldsymbol{\varepsilon}$

Correponding mathematical model. Let $z_1, \ldots, z_n$ denote assigned labels and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ locations of individual data points. Then

$$\mathrm{p}[\boldsymbol{x}_i | z_i = j, \boldsymbol{\mu}_j, \sigma] = \prod_{t=1}^{d} \frac{1}{\sqrt{2\pi}\sigma} \cdot \exp\left(-\frac{(x_{it} - \mu_{jt})^2}{2\sigma^2}\right)$$

$$\mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n | \boldsymbol{z}, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k, \sigma] = \prod_{i=1}^{n} \prod_{t=1}^{d} \frac{1}{\sqrt{2\pi}\sigma} \cdot \exp\left(-\frac{(x_{it} - \mu_{z_i t})^2}{2\sigma^2}\right)$$

Corresponding minimisation task

Again it makes sense to maximise log-likelihood instead of likelihood

$$\log \mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n | \boldsymbol{z}, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_1, \sigma] = C(\sigma) - \sum_{i=1}^{n} \sum_{t=1}^{d} \frac{(x_{it} - \mu_{z_i t})^2}{2\sigma^2}$$

The latter is equivalent to the following minimisation task:

$$\sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \boldsymbol{\mu}_{z_i} \right\|^2 \to \min$$

where the labelling $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ and cluster centres $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ is sought.

Minor problem: The optimisation task is known to be NP-hard.

Two-step minimisation algorithm

Observation. If the labels are fixed then it is easy to find optimal placement of cluster centres. First, note that we can separately optimise the location of each centre, as the sum decomposes into independent minimisation targets:

$$\sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \boldsymbol{\mu}_{z_i} \right\|^2 = \sum_{i \in \mathcal{I}_1} \left\| \boldsymbol{x}_i - \boldsymbol{\mu}_1 \right\|^2 + \cdots + \sum_{i \in \mathcal{I}_k} \left\| \boldsymbol{x}_i - \boldsymbol{\mu}_k \right\|^2$$

For each individual minimisation target F_j we can compute the derivatives

$$\frac{\partial F_j}{\partial \mu_{js}} = \sum_{i \in \mathcal{I}_j} \sum_{t=1}^{d} \frac{\partial}{\partial \mu_{js}} (x_{it} - \mu_{jt})^2 = - \sum_{i \in \mathcal{I}_j} 2(x_{is} - \mu_{is})$$

and thus $\boldsymbol{\mu}_j$ is the mean of all data points in the cluster

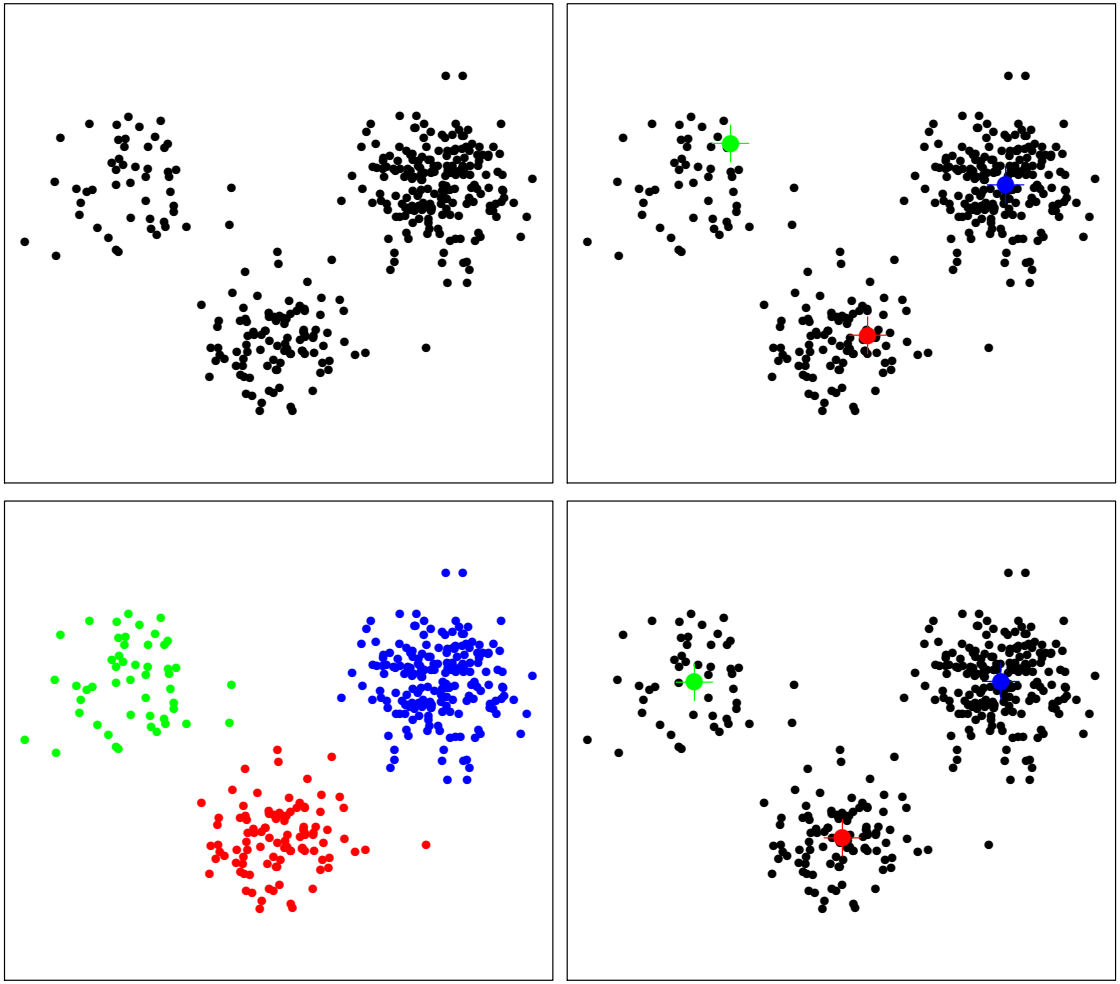
Two-step minimisation algorithm

Observation. If cluster centres are fixed then it straightforward to find best label for each data point. We must choose the closest cluster centre $\boldsymbol{\mu}_j$.

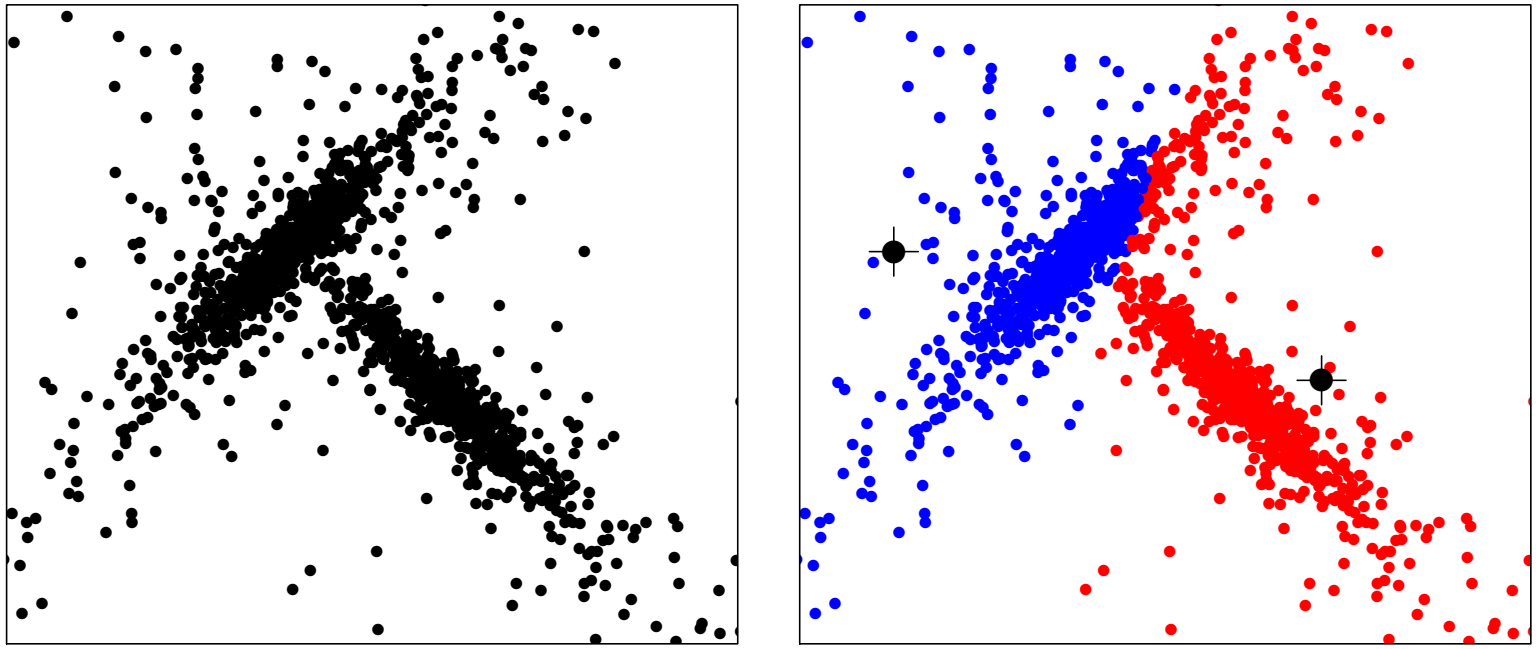
K-means clustering algorithm

1. Choose a good initial positions for cluster centres
2. For each data point $\boldsymbol{x}_i$ find the closest cluster centre $\boldsymbol{\mu}_j$ and set $z_i = j$
3. Recompute cluster centres as averages over data points belonging to it
4. Repeat from the step 2 until nothing changes.
5. Repeat with different starting points and report the best result

The algorithm really works



Example of suboptimal behaviour



K-means algorithm fails if clusters are elongated or one clusters have different density. In both cases the assumptions are not satisfied.

Model for elongated clusters

- ▷ Components of the underlying true noise are samples as $\varepsilon_i \leftarrow \mathcal{N}(0, 1)$
- ▷ Noise is reshaped using linear transformation $\boldsymbol{\varepsilon}_* = A\boldsymbol{\varepsilon}$
- ▷ The final output is generated as $\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{\varepsilon}_*$

Correponding mathematical model. It is straightforward to express the corresponding probability density function

$$\mathrm{p}[\boldsymbol{x}|\boldsymbol{\mu}, A] = \frac{1}{(\det A^T A)^{1/2}} \cdot \prod_{t=1}^{d} \frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{\varepsilon_t^2}{2}\right)$$

$$\mathrm{p}[\boldsymbol{x}|\boldsymbol{\mu}, A] = \frac{1}{(2\pi)^{d/2}(\det AA^T)^{1/2}} \cdot \exp\left(-\frac{1}{2} \cdot (\boldsymbol{x} - \boldsymbol{\mu})A^{-T}A^{-1}(\boldsymbol{x} - \boldsymbol{\mu})\right)$$

where $\Sigma = AA^T$ is known as correlation matrix

Maximum likelihood estimates

Multivariate normal distribution (*coloured Gaussian noise*) $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ is completely determines by the centre $\boldsymbol{\mu}$ and correlation matrix Σ.

It is straightforward though tedious to verify that parameter values that maximise the likelihood of data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ can be competed as follows

$$\boldsymbol{\mu} = \frac{1}{n} \cdot \sum_{i=1}^{n} \boldsymbol{x}_i$$

$$\Sigma = \frac{1}{n} \cdot X_c^T X_c$$

where X_c is a $n \times d$ matrix obtained by staking $(\boldsymbol{x}_1 - \boldsymbol{\mu})^T, \ldots, (\boldsymbol{x}_n - \boldsymbol{\mu})^T$.

Gaussian mixture model

- ▷ There are k different data sources $\mathcal{D}_1, \ldots, \mathcal{D}_k$
- ▷ Each data source has a fixed centre $\boldsymbol{\mu}_j \in \mathbb{R}^d$ and covariance matrix Σ_j
- ▷ To sample data from $\mathcal{D}_j$, we output $\boldsymbol{x} = \boldsymbol{\mu}_j + \boldsymbol{\varepsilon}$ for $\varepsilon \leftarrow \mathcal{N}(\mathbf{0}, \Sigma_j)$

Correponding hard-clustering task. Find model parameters $\boldsymbol{\Theta}$ and labels $\boldsymbol{z}$ such that the likelihood of the data is maximal:

$$\Pr\left[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n | \boldsymbol{z}, \boldsymbol{\Theta}\right] \to \max$$

Two-step minimisation algorithm

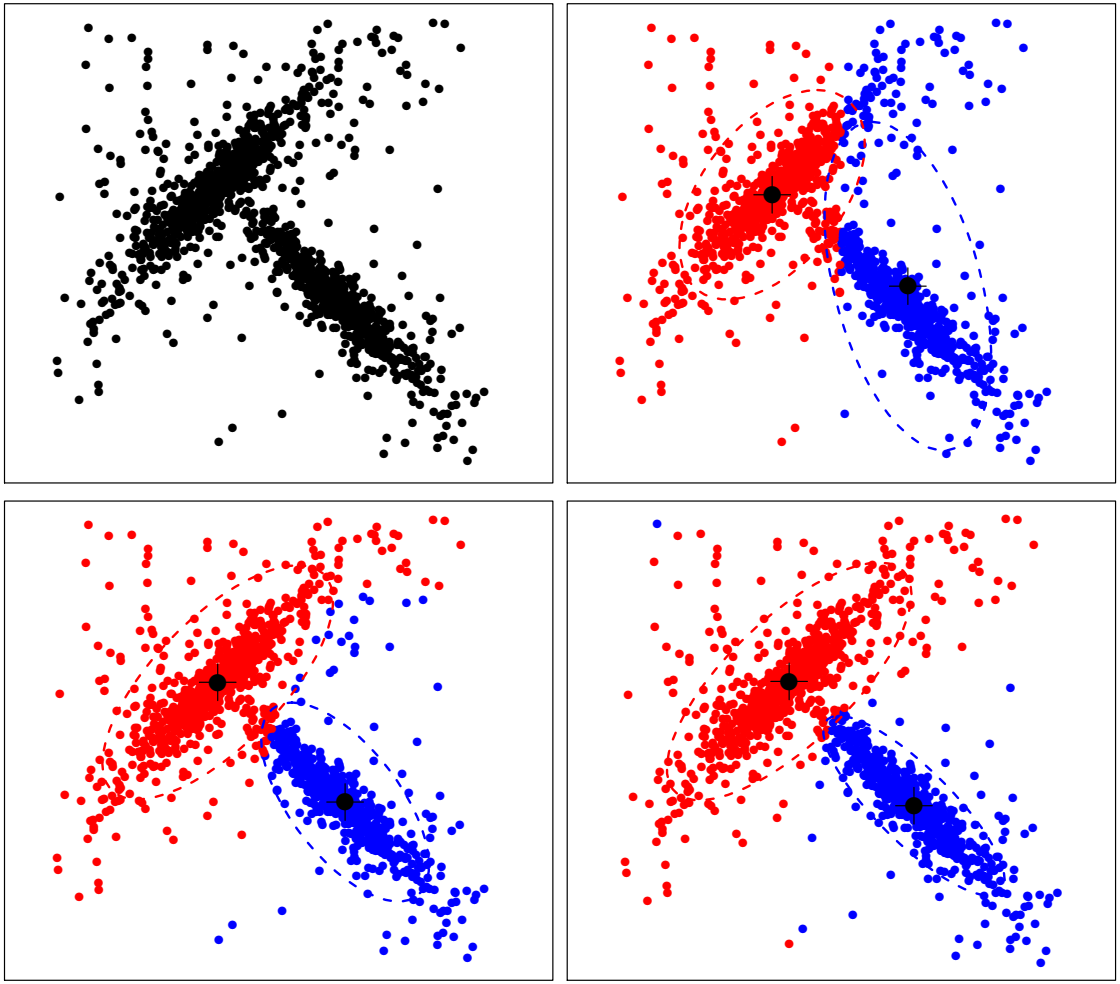
Observations

- If parameters $\boldsymbol{\Theta}$ are fixed it straightforward to find best label for each data point. We must choose z_i such that $\mathrm{p}[\boldsymbol{x}_i|\boldsymbol{\mu}_i, \Sigma_i] \to \max$.
- If labels are fixed then we can individually adjust the parameters $\boldsymbol{\mu}_j$ and Σ_j for clusters. Mixture fractions can be recomputed as $\lambda_j = |\mathcal{I}_j| / n$.

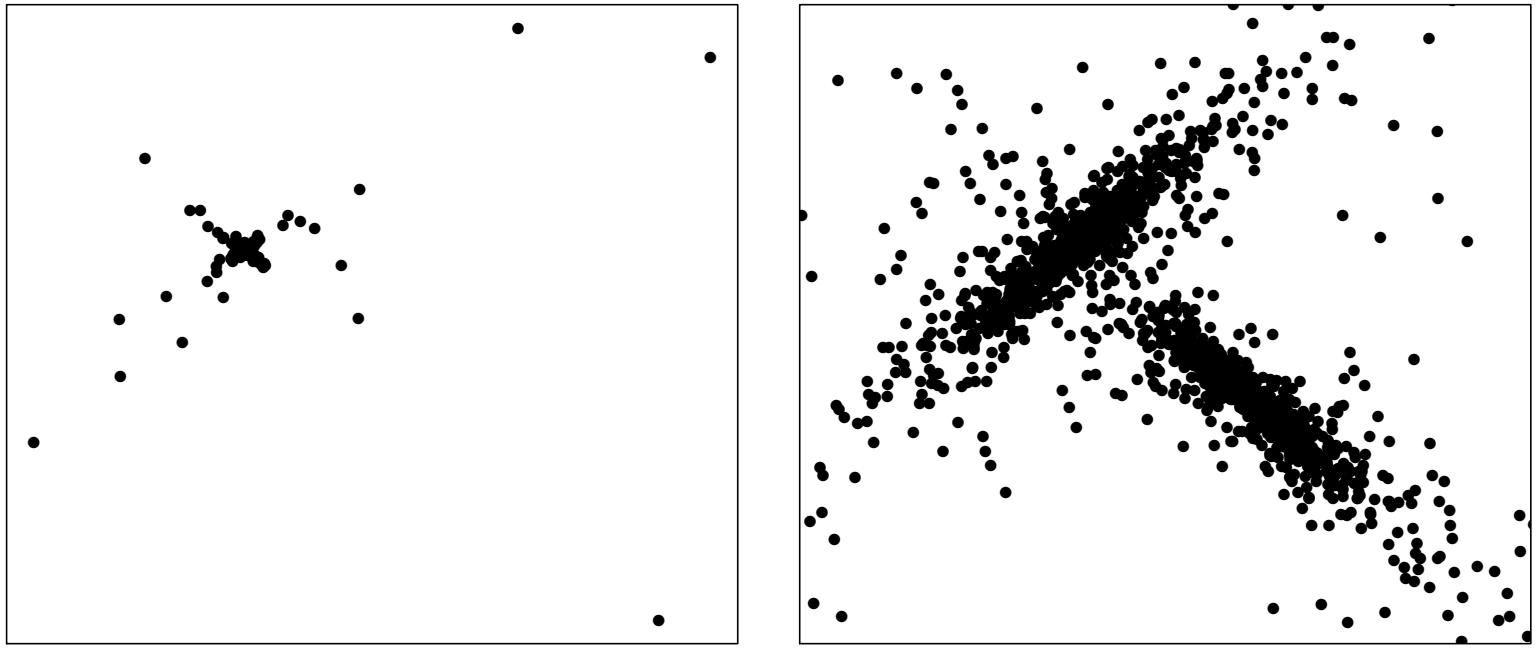
Corresponding hard-clustering algorithm

1. Choose a good initial values for cluster parameters
2. For each data point $\boldsymbol{x}_i$ find the best cluster j and set $z_i = j$
3. Recompute cluster parameters over data points belonging to it
4. Repeat from the step 2 until nothing changes
5. Repeat with different starting points and report the best result

The algorithm really works



Hard datasets for the algorithm



Few extreme values can completely offset the hard-clustering algorithm. The latter is the weakness of the Gaussian mixture model.